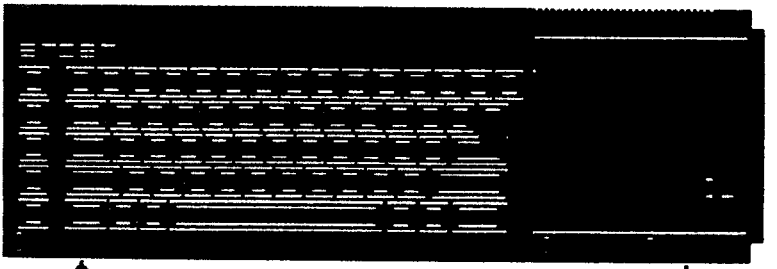
OCT.
NOV.
1989

QL_DOC

#12

$1.50



⚜ Le Bulletin Québécois ⚜ pour les Utilisateurs Sinclair QL

QL_DOC est publié à tous les 2 mois et est réalisé avec l'aide du logiciel PAGE DESIGNER II avec OMNI-DUMP de SECTOR SOFTWARE. L'impression se fait lentement mais sûrement sur une imprimante STAR NX-1000.

Vos articles, commentaires, suggestions sont appréciés.

Rédacteur: Réal Gagnon, 8286 St-Hubert, Montréal (Québec)
CANADA H2P 1Z3 (514) 381-6462

Abonnement 1 an (6 nos) : $12.00

AU SOMMAIRE

SINCLAIR NEWS

VOYAGER II.....

Le mois d'août 1989 nous a permis d'assister à un exploit technologique remarquable: la sonde spatiale VOYAGER II après un périple de 7,18 milliards de kilomètres informe les Terriens sur la nature de la planète Neptune. Le plus étonnant dans cette histoire est de constater que la sonde a été conçueavec la technologie des années 70. La caméra qui nous envoie des images de Neptune est moins perfectionnéeque les modèles courants vendus dans les magasins d'aujourd'hui. Il est assez stupéfiant d'apprendre que l'ordinateur de bord possède un mémoire vive de 38K seulement... Le QL de base en a 4 fois plus!

En avril 1990, VOYAGER sera dans la possibilité de prendre pour la première fois une photo de notre système solaire avec toutes les planètes visibles. Vers l'an 2020, la Terre perdra le contact avec la sonde. En l'an 28391, elle passera à 3,21 années-lumière, soit 30,4 millions de km, de Proxima du Centaure, l'étoile la plus proche de la Terre tandis qu'en l'an 296036, elle se rapprochera de Sirius, l'étoile la plus brillante vue de la Terre...

CLIVE frappe encore.....

L'oncle CLIVE fait parler de lui dans PC MAGAZINE.

On rapporte que lors d'une entrevue, un journaliste demande à CLIVE SINCLAIR la question suivante: Pourquoi avoir choisi un microprocesseur 8-bits (le Z80) pour votre Z88? Et Clive donna la réponse suivante: Parce que j'en ai pas trouvé de 4-bits que j'aimais!
(PC magazine, août 89, INSIDE TRACK, p. 75)

BBS SINCLAIR à TORONTO.....

Si vous avez un modem, vous pouvez contacter un babillard électronique avec une section spéciale pour les ordinateurs SINCLAIR (ZX81/2068/QL). Il s'agit du TIBM WIZARD BBS au numéro 416-743-6783. Il est en opération 24hre/par jour, 7 jours sur 7.

DANS LA BOITE A MALLE.....

Ouais, le monde des magazines SINCLAIRiens est dans le très mince. En fait, il n'y a plus qu'un seul important aux Etats-Unis, il s'agit de UPDATE MAGAZINE, l'abonnement est $18US pour un 1 an (4nos).

Içi au CANADA, à part de QL_DOC bien sûr il y a le bulletin SINC-LINK du club SINCLAIR de TORONTO. L'abonnement au bulletin est seulement de $12CAN (6nos).

UPDATE magazine;1317 Stratford Ave. Panama City FL 32404 USA
SINC-LINK;14 Richone Court, Scarborough Ontario, M1K 2Y1

HORLOGE_BAS

UTILITAIRE
SuperBASIC
par GAC-o

HORLOGE_BAS est un utilitaire qui facilite la mise à jour de l'horloge interne du QL.

Le principe du programme est simple. Avec l'aide des touches curseurs (<- et ->) et la barre ESPACE (ou la touche ENTER), on ajuste les paramètres DATE et HEURE. Une fois terminée, un fichier HORL_DAT, contenant la date et l'heure, est écri$ sur FLP1. Ce fichier sera lu, lors de la prochaine utilisation de HORLOGE_BAS. On aura alors tout simplement à ajuster le JOUR et l'heure si on le désire.

Avant d'utiliser HORLOGE_BAS, il faut créer le fichier HORL_DAT original. On peut procéder de la façon suivante:

```
10 OPEN_NEW #5,flp1_horl_dat
20 PRINT #5,"101189": REMark mois/jour/année
30 PRINT #5,"1000"  : REMark heure/minute
40 CLOSE #5
```

Si ce programme est compilé avec TURBO ou QLIBERATOR par exemple, il est facile d'insérer un ligne du genre EXEC_W FLP1_HORLOGE_EXW pour mettre l'horloge à jour dans le programme BOOT. Une fois terminé , le programme BOOT poursuit son exécution.

```
) HORLOGE_BAS par RG-089 pour QL_DOC

100 REMark mise à jour de l'horloge QLienne
110 REMark par Réal Gagnon  MTL pour QL_DOC
120 REMark Logiciel Domaine Publique pour le SINCLAIR QL
130 :
140 LET dev$="flp1_"
150 OPEN#7,con 260x110a120x50
160 PAPER#7,7,2,2:CSIZE #7,2,0:INK#7,2:BORDER#7,2,7:CLS#7
170 PRINT#7," * HORLOGE v1 RG89 * ":INK#7,7
175 OVER#7,1:CURSOR#7,1,1:PRINT#7," * HORLOGE v1 RG89 *":INK#7,0
177 CURSOR#7,0,2:UNDER#7,1:PRINT#7,"                    ":OVER#7,0
180 UNDER#7,0:PRINT #7,"<- & -> "\" changent la valeur"
190 PRINT #7,"[ENTER] ou [ESPACE]"\" valident la valeur"
200 :
210 lecture_horl_dat
220 modifie_date      : modifie_heure
230 mise_a_jour_du_QL : ecriture_horl_dat
240 :
```

```
250 DEFine PROCedure lecture_horl_dat
260   OPEN #8,dev$&"horl_dat"
270   INPUT#8,anc_dat$;anc_heu$
280   CLOSE#8
290 END DEFine
300 :
310 DEFine PROCedure modifie_date
320 s$=" - " : AT#7,6,0 : PRINT #7,"DATE   (mm - jj - aa)"
330 PRINT #7,TO 8,anc_dat$(1 TO 2)&s$&anc_dat$(3 TO 4)&s$&anc_dat$(5 TO)
340 n_dat$=entree_val$(anc_dat$,3,7,12,31,99,1)
350 END DEFine
360 :
370 DEFine PROCedure modifie_heure
380 AT#7,8,0 : PRINT#7,"HEURE  (hh : mm)"
390 PRINT#7,TO 8,anc_heu$(1 TO 2)&" : "&anc_heu$(3 TO 4)
400 n_heu$=entree_val$(anc_heu$,2,9,24,59,0,0)
410 END DEFine
420 :
430 DEFine PROCedure mise_a_jour_du_QL
440 SDATE "19"&n_dat$(5 TO 6),n_dat$(1 TO 2),n_dat$(3 TO 4),
    n_heu$(1 TO 2),n_heu$(3 TO 4),0
450 END DEFine
460 :
470 DEFine PROCedure ecriture_horl_dat
480   DELETE dev$&"horl_dat"
490   OPEN_NEW #8,dev$&"horl_dat"
500   PRINT#8,n_dat$\n_heu$
510   CLOSE#8
520 END DEFine
530 :
540 DEFine FuNction entree_val$ (orig$,combien,ligne,lim1,lim2,lim3,min)
550 FOR boucle=0 TO combien-1
560   colonne=8+(boucle*5): AT#7,ligne,colonne : PAPER#7,7
570   xxx=orig$(((boucle*2)+1)TO(boucle*2)+2)
580   IF xxx<10:PRINT#7, "0";
590   PRINT #7,xxx
600   REPeat loop
610    touche=CODE(INKEY$(#7,-1))
620    SELect ON touche
630     =192
640        IF xxx>min : xxx=xxx-1
650     =200
660        SELect ON boucle
670          =0: IF xxx<lim1 : xxx=xxx+1
680          =1: IF xxx<lim2 : xxx=xxx+1
690          =2: IF xxx<lim3 : xxx=xxx+1
700        END SELect
710     =10,32: BEEP 5,100: EXIT loop
720     END SELect
730    AT#7,ligne,colonne
740    IF xxx<10:PRINT#7,"0";
750    PRINT#7,xxx
760   END REPeat loop
770   IF xxx<10
780      orig$(((boucle*2)+1)TO(boucle*2)+2)="0"&xxx
790    ELSE
800      orig$(((boucle*2)+1)TO(boucle*2)+2)=xxx
810    END IF
820 END FOR boucle
830 PAPER#7,7,2,2 : RETurn orig$
840 END DEFine
```

ETAT_BIN

SuperBASIC Extension
par W. Lenerz
(QLCF)

Ce nouveau mot-clé S-Basic peut vous aider dans la manipulation de vos fichiers. En effet, un des désavantages du S-Basic est que, lorsqu'on veut faire une opération sur un fichier (copie, ouverture etc...) et que celui-ci n'existe pas, le S-Basic s'arrête avec une erreur. Ce n'est pas grave lorsqu'on travaille en mode directe, mais c'est embêtant dans un programme, car l'erreur va arrêter le programme. ETAT est une nouvelle fonction qui vous permettra de tester l'état d'un fichier avant de faire une opération dessus.

Le nouveau mot-clé se charge par:
a=RESPR(200): LBYTES <unité>_etat_bin,a:CALL a

Vous disposez à ce moment d'une nouvelle fonction, appelée ETAT.

Syntaxe: ETAT ("nom_du_fichier"); où nom_du_fichier doit être le nom du fichier à tester, y compris le nom de l'unité microdrive, disquette ou randisque. Ce nom doit être compris entre guillemets et entre parenthèses. Cependant, on peut utiliser aussi un nom de variable, qui contient le nom du fichier. Dans ce cas, il ne faut plus mettre les guillemets - mais il faut toujours mettre les parenthèses (v. exemple 2). Le nouveau mot-clé ETAT est une fonction. Il faut donc l'utiliser dans des lignes du genre: PRINT ETAT ('nom du fichier') ou LET a=ETAT('Nom du fichier'). La fonction ETAT vous renverra un code d'erreur. Si ce code est égal à 0, le fichier en question existe sur l'unité en question. Sinon, la valeur renvoyée par ETAT correspond aux différents codes d'erreur du QDOS. Vous trouverez dans le manuel du QL (section 'CONCEPTS' - titre 'Error Handling') le sens de chacun des codes d'erreur (négatifs) renvoyés.

EXEMPLES D'UTILISATION
Exemple.1 Utilisation en mode direct:

```
PRINT ETAT('mdv1_coucou')
```

Ceci va imprimer un chiffre à l'écran, qui correspond au code d'erreur.

Exemple 2: On demande le nom d'un fichier à copier, source et destination:

```
10 INPUT "Vous voulez copier quel fichier ?";a$
20 testa=ETAT(a$)
30 IF testa<>0 THEN PRINT "Ce fichier n'existe pas!":
   GOTO 10
40 INPUT "Copier vers quel fichier?";b$
50 testb=ETAT(b$)
60 IF testb<>-7 THEN PRINT "Ce fichier existe déjà ou
   est en usage!":GOTO 40
70 COPY a$ TO b$
```

ETAT_BIN de W. Lenerz (QLCF)

```
*******************************
*       ETAT_BIN              *
*        de W. Lenerz         *
*          source QLCF        *
*******************************
debut   MOVE.W    $110,A2       Vecteur pour lier mots dans la liste
        LEA       procs,A1      des mots S-basic
        JMP       (A2)          A1 pointe vers le mots-clé
procs   DC.W      0              nombre de procédures
        DC.W      0              fin des procédures
        DC.W      1              nombre de fonctions
        DC.W      etat-*         pointeur vers début de la routine
        DC.B      4              longueur du nom
        DC.B      'ETAT'         nom
        DC.L      0              fin du bloc de définition
etat    MOVEA.W $0116,A0        vecteur pour chercher un paramètre alpha, qui
        JSR       (A0)          est mis sur le pile, avec un mot indiquant
        TST.L     D0            sa longueur. Ce paramètre, c'est le nom du fichier
        BNE.S     sortie        si erreur: on sort
        CMPI.W    #1,D3         D3 contient le nombre de paramètres
        BNE.S     sortie        si <> de 1, il y a une erreur: on sort
        MOVE.L    A1,A2         A1 (et maintenant A2) pointe vers la pile, par rapport
        ADDA.L    A6,A2         à A6. En additionnant A6, A2 en devient indépendant
        MOVE.W    (A2)+,D2      D2 contient le mot indiquant sa longueur
        LEA       buffer,A1
        MOVE.W    D2,(A1)+      et on le transfère dans le buffer
        SUBQ.W    #1,D2
boucle  MOVE.B    (A2)+,(A1)+
        DBF       D2,boucle     puis on transfère le nom du fichier dans le
        MOVEQ     #0,D3         buffer, qu'on va utiliser pour ouvrir
        MOVEQ     #-1,D1        le fichier
        LEA       buffer,A0
        MOVEQ     #1,D0
        TRAP      #2            et on tente d'ouvrir le fichier
        TST.L     D0
        BNE.S     sortie
        MOVEQ     #2,D0         on referme le fichier
        TRAP      #2
sortie  MOVEA.L $58(A6),A1          $58(A6) contient la pile S-Basic
        SUBQ.W    #2,A1             on se fait de la place
        MOVE.L    A1,$005B(A6)      et on y met le paramètre de retour,
        MOVE.W    D0,$00(A6,A1.L)   à savoir le code d'erreur contenu dans D0
        MOVEQ     #$03,D4           on signale qu'on va retourner un entier
        MOVEQ     #$00,D0
        RTS                         et on y va
buffer  DS.L 10
        END
```

Codes retournés par la fonction ETAT	
0	Opération OK
-7	Fichier n'existe pas (not found)
-8	Fichier existe déjà (already exists)
-9	Fichier non-disponible (in use)
-11	Lecteur plein (drive full)
-16	Erreur d'opération (bad or changed medium)

Chargeur BASIC universel

```
100 REMark CHARGEUR_BAS
110 REMark pour QL_DOC
120 :
130 INPUT "Où sauver les codes (ex. flp1_)?";sauve$
140 IF LEN(sauve$)=0:sauve$="flp1_"
150 PRINT "Un instant s.v.p.";
160 :
170 RESTORE 1000
180 memoire=RESPR(1000):base_memoire=memoire
190 compte=0
200 :
210 REPeat boucle
220   IF EOF:EXIT boucle
230   READ octet:POKE memoire,octet:memoire=memoire+1
240   compte=compte+1
250   PRINT ".";
260 END REPeat boucle

( insérer içi les DATA nécessaires )
```

DATA pour la création de ETAT_BIN

```
900 PRINT\"Une touche pour sauver "&sauve$&"ETAT_BIN "
910 PRINT "("&compte&" octets)"
920 PAUSE
930 SBYTES sauve$&"etat_bin",base_memoire,compte
940 PRINT "Ok!"
950 :
1000 DATA  52, 120,   1,  16,  67, 250,   0,   4,  78, 210
1001 DATA   0,   0,   0,   0,   0,   1,   0,  12,   4,  69
1002 DATA  84,  65,  84,   0,   0,   0,   0,   0,  48, 120
1003 DATA   1,  22,  78, 144,  74, 128, 102,  46,  12,  67
1004 DATA   0,   1, 102,  40,  36,  73, 213, 206,  52,  26
1005 DATA  67, 250,   0,  52,  50, 194,  83,  66,  18, 218
1006 DATA  81, 202, 255, 252, 118,   0, 114, 255,  65, 250
1007 DATA   0,  34, 112,   1,  78,  66,  74, 128, 102,   4
1008 DATA 112,   2,  78,  66,  34, 110,   0,  88,  85,  73
1009 DATA  45,  73,   0,  88,  61, 128, 152,   0, 120,   3
1010 DATA 112,   0,  78, 117,   0,   0,   0,   0,   0,   0
1011 DATA   0,   0,   0,   0,   0,   0,   0,   0,   0,   0
```

ABACUS
Application

- Classement de fichiers -

Dans le QL_DOC#11, on présentait une nouvelle version du programme DLO_BAS qui sert à classer alphabétiquement les contenusd'une disquette ou d'un µdv. Nous allons voir une méthode différente pour faire le même travail avec ABACUS.

Premièrement il nous faut un petit programme SuperBASIC pour lire le DIRectory et le transformer sous une forme qui peut être IMPORTer avec ABACUS.

La façon de procéder est simple. On crée notre fichier contenant le DIRectory ("direct_tmp") de façon standard. Puis on relie ce fichier en séparant chacun de ses éléments de façon à ce qu'ils soient acceptés par la fonction IMPORT d'ABACUS.

Pour importer du texte dans ABACUS, notre texte doit se présenter de la façon suivante: "TEXTE 1","TEXTE 2","TEXTE 3", etc... . Chaque chaîne de caractères est entourée de guillemets et séparée par une virgule. Notre fichier séparé ainsi se nommera DIRECT_EXP.

Dans ABACUS, nous insérons la disquette/cartouche contenant le fichier DIRECT_EXP.

Nous faisons F3-F-I-direct_exp-(ENTER)-(ENTER).

Le contenu devrait apparaître à l'écran. La prochaine étape consiste à classer par ordre alphabétique le nom de fichier avec la commande ORDER.

Nous faisons F3-O-(ENTER)-(ENTER)-(ENTER).

L'étape suivante consiste à utiliser la commande COPY pour mettre nos fichiers en colonnes. A ce moment, notre imagination est libre de placer comme bon il nous semble les colonnes.

Par exemple, si nous avons 20 fichiers, on pourrait faire 4 colonnes de 5 fichiers. On importe le fichier DIRECT_EXP dans ABACUS, on fait le tri alphabétique (SORT). Les 2 premières lignes sont pour le nom de la disquette/µdv et le nombre de secteurs utilisés. Les rangées 3 à 22 contiennent les noms de fichiers. Avec la commande COPY, nous allons faire 4 colonnes de 5 fichiers.

F3-C-A3:A7(ENTER)-B3(ENTER)
F3-C-A8:A12(ENTER)-C3(ENTER)
F3-C-A13:17(ENTER)-D3(ENTER)
F3-C-A18:22(ENTER)-E3(ENTER)

Il est possible de changer la largeur des cellules si celles-ci sont trop étroites pour la longueur des noms de fichiers, avec l'option WIDTH de la commande GRID.

Un fois ceci fait il ne reste plus qu'à effacer le surplus qui reste dans la colonne A avec la commande RUBOUT A3:A22.

> ABACUS Application par RG-089

```
100 REMark DIRECTORY SORT ABACUS V1
110 REMark par Réal Gagnon
120 :
130 REPeat debut
140 INPUT 'Le directory de quel DEVICE? (ex. flp1_) ',,medium$
150 IF medium$='':PRINT 'Retour au SuperBASIC.':STOP
160 IF medium$(1 TO 3) INSTR 'mdvflpfdkhdk':EXIT debut
170 END REPeat debut
180 :
190     DELETE  medium$&'direct_EXP'
200     DELETE  medium$&'direct_TMP'
210 OPEN_NEW #6,medium$&'direct_TMP'
220     DIR  #6,medium$
230     CLOSE#6
240 :
250      OPEN#6,medium$&'direct_TMP'
260  OPEN_NEW#7,medium$&'direct_EXP'
270  PRINT "Mmmm";
280 REPeat BOUCLE
290    INPUT#6,FICHIER$
300    PRINT ".";
310    IF FICHIER$=='"direct_tmp"'
320      END REPeat BOUCLE
330       END IF
340    IF EOF(#6)
350      CLOSE#6:CLOSE#7
360       EXIT BOUCLE
370        END IF
380    PRINT#7,'"';FICHIER$;'"';',';
390 END REPeat BOUCLE
400 :
410 PRINT
420 PRINT "OK Terminé!"
```

MAJI_EXE

Un programme multi-tâche par W. Lenerz (QLCF)

MAJI_EXE est un petit logiciel supposé tourner en multitâche avec QUILL. MAJI_EXE va afficher dans la zone d'état de QUILL le mot "Maj.:". Si vous êtes en mode majuscule, il affichera deux flèches vers le haut à côté de ces mots. MAJI_EXE va aussi afficher une horloge permanente dans la zone d'état, là où elle dérange le moins.

Dans votre programme BOOT de QUILL, vous insérerez une ligne du style 'EXEC <unité>_MAJI_EXE' juste avant la ligne "EXEC_W <unité>_QUILL" (ou "EXEC_W <unité>_QLWP"). Le programme MAJI_EXE doit être sur l'unité, bien sûr. MAJIc_EXE (pour MAJI court) est une version racourcie de MAJI, qui n'affiche pas d'horloge et qui n'affiche que le mot "Maj.:" suivi ou non de flèches.

Que ceux qui ont une unité de disquettes ne s'affolent pas: MAJI_EXE et MAJIc_EXE mettent un certain temps avant d'apparaître sur l'écran. Ceci a été fait exprès, pour que MAJI_EXE n'apparaisse pas avant que QUILL ne soit chargé. Avec des mdvs, QUILL met dans les 8 secondes à s'installer, MAJI_EXE se suspend donc au début pendant 8 secondes, afin de permettre à QUILL de s'installer. Ceux qui ont des disquettes peuvent varier le temps que met MAJI_EXE à apparaître, en tapant les commandes suivantes (faire ça sur une copie de MAJI_EXE, pas sur l'original):

```
10 a = RESPR(234)
20 LBYTES <unité>_MAJI_EXE, a
30 POKE_W a+30,xxx
40 DELETE <unité>_MAJI_EXE
50 SEXEC <unité>_MAJI_EXE,a,234,50
```

où xxx est un chiffre représentant xxx 60ièmes de secondes. Par conséquent, si vous voulez suspendre MAJI pendant 3 secondes, xxx doit être égal à 180.

Pour MAJIc_EXE cela devient:

```
10 a = RESPR(156)
20 LBYTES <unité>_MAJIc_EXE, a
30 POKE_W a+30,xxx
40 DELETE <unité>_MAJIc_EXE
50 SEXEC <unité>_MAJIc_EXE,a,156,20
```

Ne POKE_W pas des valeurs supérieures à 65536, ou inférieures à 0...

/* L'idée de ce programme m'a été inspiré par un programme nommé 'Capsclock' que j'ai trouvé sur la logithèque QUANTA. Je me suis fortement inspiré de ce programme, notamment pour le positionnement des fenêtres, mais j'ai écrit le code moi-même. */

MAJI_EXE par W. Lenerz (QLCF)

```
;;;;;;;;;;;;;;;;;;;;;;;;;;
;      MAJI_EXE          ;
;    par W. Lenerz       ;
;    source QLCF         ;
;;;;;;;;;;;;;;;;;;;;;;;;;;
       BRA.S   debut              saut au début du job
       DC.L    'Wolf'             entête standard
       DC.W    $4AFB
       DC.W    4
       DC.W    'Maji'             nom du job

debut  MOVEQ   #11,D0
       MOVEQ   #-01,D1            TRAP 1,D0=11 ($0B hex): changement de
       MOVEQ   #01,D2             priorité du job. D2 contient la nou-
       TRAP    #01                velle priorité:1 (la plus basse)
```

```
        MOVEQ   #08,D0
        MOVEQ   #-01,D1          TRAP 1, D0=8 :on suspend le job pen-
        MOVE.W  #400,D3          dant 400/60è de secondes pour permet-
        SUBA.L  A1,A1            tre à Quill de se charger. Ceux avec
        TRAP    #$01             des disquettes peuvent raccourcir ça
        MOVEA.W $00C8,A2
        LEA     fen1,A1          établissement de la 1ère fenetre
        JSR     (A2)
        MOVEA.L A0,A5            A5=ID canal 1ère fen.
        MOVEA.W $00C8,A2
        LEA     fen2,A1
        JSR     (A2)             établissement de la 2è fen.
        MOVEA.L A0,A4            A4=ID canal 2ème fen.
        SUBA.L  A6,A6
boucle  MOVEA.L A4,A0
        MOVEQ   #$11,D0          TRAP 3, D0=$11: positionnement du
        MOVEQ   #$00,D1          curseur dans la 2è fen.
        MOVEQ   #-$01,D3         D1= No. de colonne
        TRAP    #$03
        MOVEQ   #$07,D0
        MOVEQ   #$06,D2          TRAP 3, D0=7: envoi d'une chaîne
        MOVEQ   #-$01,D3         d'octets. D2 contient le nombre d'oc-
        LEA     maj,A1           tets à envoyer, A0 l'ID du canal
        TRAP    #$03
        CMP.B   $00028088,D0     si contenu de 28088h =D0 (=0) alors
        BEQ.S   suppri           on est en minuscules et on saute
        MOVEQ   #$29,D0          à 'supprime'. Sinon on affiche 'ﻍﻍ'
        MOVEQ   #$07,D1          TRAP 3, D0=$29:
        MOVEQ   #-$01,D3         On met l'encre en blanc  (D1)
        TRAP    #$03
        MOVEQ   #$07,D0          On envoit 'ﻍﻍ', TRAP 3, D0=7
        MOVEQ   #$02,D2          A1 contient l'adresse du début de
        MOVEQ   #-$01,D3         la chaîne
        LEA     fleche,A1
        TRAP    #$03

        MOVEQ   #$29,D0
        MOVEQ   #$04,D1          On remet l'encre au vert
        MOVEQ   #-$01,D3
        TRAP    #$03
        BRA.S   horlo            et on saute à l'horloge
suppri  MOVEQ   #$07,D0          on supprime la flèche en imprimant
        MOVEQ   #$02,D2          deux espaces
        MOVEQ   #-$01,D3
        LEA     espace,A1
        TRAP    #$03
horlo   MOVEQ   #$13,D0          début de l'horloge
        TRAP    #$01             on cherche l'heure, ds D1
*                                 maintenant, on convertit la date
*                                 dans D1 en ASCII
        MOVEA.W $00EC,A2         vecteur concerné
        LEA     pile,A1          debut de la pile qui est à l'envers
        JSR     (A2)             et on convertit la date (ds D1)
        SUB.L   D2,D2
        LEA     pile,A1          debut de la pile
        SUB.W   #22,A1
        MOVEA.L A5,A0
        MOVEQ   #$07,D0          puis on affiche la date
        MOVE.W  (A1)+,D2
        MOVEQ   #-$01,D3
        TRAP    #$03
        MOVEQ   #$11,D0          et on positionne le curseur
        MOVEQ   #$00,D1          D1 = col. curseur
        MOVEQ   #-$01,D3
```

```
        TRAP    #$03
        MOVEQ   #$0B,D0          on va suspendre le job
        MOVEQ   #-$01,D1         pendant 5/60 sec
        MOVEQ   #$05,D3
        SUBA.L  A1,A1
        TRAP    #$01
        BRA.S   boucle
*
fen1    DC.B    0                Définition de la fenetre
        DC.B    0                  couleur bord
        DC.B    0                  largeur bord
        DC.B    7                  couleur papier
        DC.W    120                couleur encre
        DC.W    10                 largeur
        DC.W    162                hauteur
        DC.W    246                coord. x du coin gauche sup.
fen2    DC.B    0                     "    y  "  "     "     "
        DC.B    0                  définition fenetre 2
        DC.B    0                  comme ci-haut...
        DC.B    4
        DC.W    54
        DC.W    10
        DC.W    114
        DC.W    236
maj     DC.B    'Maj.: '
fleche  DC.W    '↑↑'
espace  DC.W    '  '
        DC.L    0
        DC.L    0,0,0,0,0
prie    DC.L    0
pseudo  DC.L    0
        END
```

› Chargeur universel SuperBASIC

```
100 REMark CHARGEUR_BAS
110 REMark pour QL_DOC
120 :
130 INPUT "Où sauver les codes (ex. flp1_)?";sauve$
140 IF LEN(sauve$)=0:sauve$="flp1_"
150 PRINT "Un instant s.v.p.";
160 :
170 RESTORE 1000
180 memoire=RESPR(1000):base_memoire=memoire
190 compte=0
200 :
210 REPeat boucle
220   IF EOF:EXIT boucle
230   READ octet:POKE memoire,octet:memoire=memoire+1
240   compte=compte+1
250   PRINT ".";
260 END REPeat boucle
```

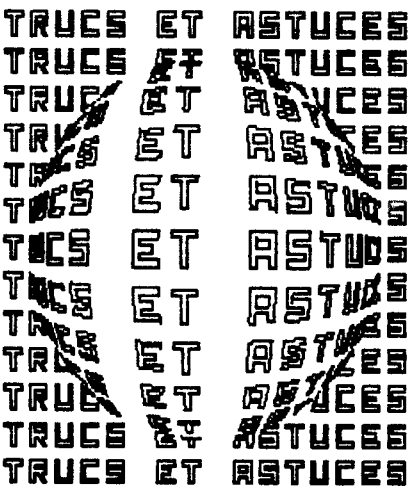
› Insérez içi les lignes DATA (

DATA pour la création de MAJI_EXE

```
900 PRINT \"Une touche pour sauver "&sauve$&"MAJI_EXE"
910 PRINT "("&compte&" octets)"
915 PAUSE
920 SEXEC sauve$&"maji_exe",base_memoire,compte,50
925 PRINT "Ok!"
930 :
1000 DATA  96,  12,  87, 111, 108, 102,  74, 251,   0,   4
1001 DATA  77,  97, 106, 105, 112,  11, 114, 255, 116,   1
1002 DATA  78,  65, 112,   8, 114, 255,  54,  60,   1, 144
1003 DATA 147, 201,  78,  65,  52, 120,   0, 200,  67, 250
1004 DATA   0, 146,  78, 146,  42,  72,  52, 120,   0, 200
1005 DATA  67, 250,   0, 146,  78, 146,  40,  72, 157, 206
1006 DATA  32,  76, 112,  17, 114,   0, 118, 255,  78,  67
1007 DATA 112,   7, 116,   6, 118, 255,  67, 250,   0, 132
1008 DATA  78,  67, 176,  57,   0,   2, 128, 136, 103,  30
1009 DATA 112,  41, 114,   7, 118, 255,  78,  67, 112,   7
1010 DATA 116,   2, 118, 255,  67, 250,   0, 110,  78,  67
1011 DATA 112,  41, 114,   4, 118, 255,  78,  67,  96,  12
1012 DATA 112,   7, 116,   2, 118, 255,  67, 250,   0,  90
1013 DATA  78,  67, 112,  19,  78,  65,  52, 120,   0, 236
1014 DATA  67, 250,   0, 102,  78, 146, 148, 130,  67, 250
1015 DATA   0,  94, 146, 252,   0,  22,  32,  77, 112,   7
1016 DATA  52,  25, 118, 255,  78,  67, 112,  17, 114,   0
1017 DATA 118, 255,  78,  67, 112,   8, 114, 255, 118,   5
1018 DATA 147, 201,  78,  65,  96, 130,   0,   0,   0,   7
1019 DATA   0, 120,   0,  10,   0, 162,   0, 246,   0,   0
1020 DATA   0,   4,   0,  54,   0,  10,   0, 114,   0, 236
1021 DATA  77,  97, 106,  46,  58,  32, 190, 190,  32,  32
1022 DATA   0,   0,   0,   0,   0,   0,   0,   0,   0,   0
1023 DATA   0,   0,   0,   0,   0,   0,   0,   0,   0,   0
1024 DATA   0,   0,   0,   0,   0,   0,   0,   0,   0,   0
1025 DATA   0,   0,   0,   0,   0,   0,   0,   0,   0,   0
```

DATA pour la création de MAJIC_EXE

```
900  PRINT \"Une touche pour sauver "&sauve$&"MAJIC_EXE"
910  PRINT "("&compte&" octets)"
920  PAUSE
930  SEXEC sauve$&"MAJIC_EXE",base_memoire,compte,20
940  PRINT "Ok!"
950 :
1000 DATA  96,  14,  87, 111, 108, 102,  74, 251,   0,   5
1001 DATA  77,  97, 106, 105,  99,   0, 112,  11, 114, 255
1002 DATA 116,   1,  78,  65, 112,   8, 114, 255,  54,  60
1003 DATA   1, 194, 147, 201,  78,  65,  52, 120,   0, 200
1004 DATA  67, 250,   0,  82,  78, 146,  42,  72, 157, 206
1005 DATA  32,  77, 112,  17, 114,   0, 118, 255,  78,  67
1006 DATA 112,   7, 116,   6, 118, 255,  67, 250,   0,  68
1007 DATA  78,  67, 176,  57,   0,   2, 128, 136, 103,  30
1008 DATA 112,  41, 114,   7, 118, 255,  78,  67, 112,   7
1009 DATA 116,   2, 118, 255,  67, 250,   0,  46,  78,  67
1010 DATA 112,  41, 114,   4, 118, 255,  78,  67,  96, 196
1011 DATA 112,   7, 116,   2, 118, 255,  67, 250,   0,  26
1012 DATA  78,  67,  96, 182,   0,   0,   0,   4,   0,  54
1013 DATA   0,  10,   0, 114,   0, 236,  77,  97, 106,  46
1014 DATA  58,  32, 190, 190,  32,  32
```



Voici 2 combinaisons de touches non-documentées présentes dans le programme graphique EASEL.

SHIFT F3 - efface les lignes commandes du bas de l'écran libérant ainsi tout l'espace pour le graphique.

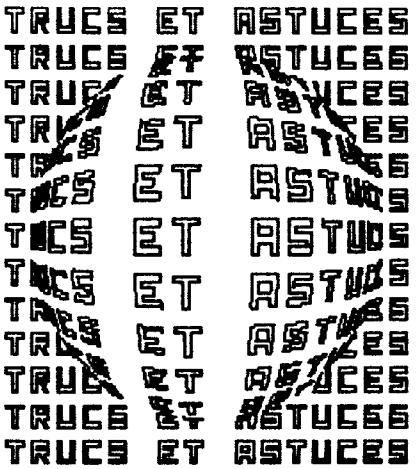
SHIFT F4 - permet de sélectionner un couleur de fond pour le graphique.
A la question : Please enter backdrop colour number?
On répond avec un chiffre de 0 à 7.

Quoi de neuf au programme?

Digital Precision annonce un nouvelle version de LIGHTNING. Elle a pour nom LIGHTNING SPECIAL EDITION selon une tradition chère à DP. Jusqu'à 30% plus rapide que la version originale, LIGHTNING SE se présente sur la forme ROM pour obtenir le maximum de rapidité. Une version PROGRAMME est également disponible.
LIGHTNING SE ... £50 LIGHNING régulier ... £25

Le traitement de texte ultra-performant TEXT 87 2.00 est maintenant disponible en version francaise auprès de son éditeur SOFTWARE 87. Le prix est de £45.
SOFTWARE 87,33 Savernake Road, London NW3 2JU, England

Un compilateur C++ est sur le point de faire son apparition. Il sera produit par la compagnie danoise DANSOFT, la même qui produit le THOR XVI, un ordinateur compatible QL.
DANSOFT, PO BOX 59, DK 1002 Copenhagen K, Denmark



Le VIDE n'est pas égale à ZERO...

ABACUS considère qu'une cellule vide possède une valeur numérique de zéro ("0"). Pour le prouver, faisons le petit exercice suivant:

[A1] "Cellule test A2"
[C2] "Valeur de la cellule A2 ->"
[F2] A2

Bon, on remarque que la formule dans F2 retourne "0" même si A2 est vide ("EMPTY"). Si le curseur est en A2, on peut entrer des valeurs numériques et vérifier que la cellule F2 reflète les changements de valeur de A2. Si on efface totalement A2 avec RUBOUT A2:A2, alors F2 reprend la valeur de "0".

Pour corriger ce petit problème, nous allons utiliser la fonction COUNT qui elle ne prend en ligne de compte les cellules vides. On remplace la formule en F2 par celle-çi: IF(COUNT(A2:A2),A2,""). Cette formule donnera la valeur de A2 seulement si cette dernière est non-VIDE, sinon elle retourne un chaîne de caractère vide.

La mémoire-écran du QL débute à l'adresse 131072 et occupe 32K. Pour sauver le contenu de l'écran, nous pouvons faire la commande SBYTES mdv1_ecran,131072,32768.

Mais ces chiffres sont difficiles à retenir. Pour nous faciliter la tâche, nous pouvons utiliser cet équivalent plus facile à se rappeler: SBYTES mdv1_ecran,2^17,2^15.

Et pour charger et visualiser un écran, nous pouvons faire
la commande : LBYTES mdv1_ecran,2^17 ou
LBYTES mdv1_ecran,128*1024

Logiciels
QL_DOC

QL_DOC offre à ses lecteurs la possibilité d'obtenir des programmes faisant partie du domaine public.

Le seul coût est celui des frais postaux.

Le lecteur DOIT fournir le support sur lequel il veut recevoir les programmes. C'est-à-dire que VOUS devez fournir les micro-cartouches ou les disquettes.

#1 Disquette QL_DOC-1
.CLAVIER_EXE,ABACLAV_EXE,TRA2_BAS
HORLOGE_BAS,DLO_BAS,etc...

#2 QLINK 1.556 .programme de communication
(partagiciel/shareware)

#3 LE GRAPHISTE .programme de dessin
à la EYE-Q

#4 QL-IBM .lecture/écriture disquette MSDOS

#5 JEUX-1
.Breakout,Starport 2001,M-Cruncher,Snake

#6 UTILITE-1
.CST utilities, QLSUB Multi-Boot, Convert

#7 QLCF-1 .Domino,Taquin,Othello, etc...

Prix : $ 4,00 chaque